

DEC 28 2006

Application No. 10/615,065  
Amendment dated December 28, 2006  
Reply to Office Action of September 28, 2006

Docket No.: 4444-0294PUS1

**AMENDMENTS TO THE CLAIMS**

1. (Currently Amended) A method for implementing a modular exponentiation in a cryptographic operation on a digital message in a computer system, the modular exponentiation involving a secret exponent key, protecting public key schemes from timing, power monitoring and fault attacks the method comprising the steps of:

~~obtaining a message for use in a cryptographic operation;~~

~~obtaining a modulus and an exponent corresponding to said cryptographic operation, wherein said exponent contains at least one bit;~~

~~initializing a first value as a one, and assigning said message to a second value;~~

~~executing a modulo exponentiation algorithm on each bit of said exponent from the most significant bit to the least significant bit, wherein said algorithm comprising the steps of:~~

~~input a bit to an inverter and storing the output as a third value, and assigning the next bit of said bit as a fourth bit;~~

~~if said third value is a zero, updating said first value with the result of squaring, modulo said modulus said first value, if said third value is a one, updating said first value with the result of multiplying, modulo said modulus said first value by said second value; and~~

~~if said fourth value is a zero, updating said first value with the result of squaring, modulo said modulus said first value, if said fourth value is a one, updating said first value with the result of multiplying, modulo said modulus said first value by said second value;~~

~~updating said bit with the next bit of said bit, and executing steps of said algorithm on said bit until said bit being said least significant bit of said exponent; and~~

~~storing and output said first value~~

Application No. 10/615,065  
Amendment dated December 28, 2006  
Reply to Office Action of September 28, 2006

Docket No.: 4444-0294PUS1

unconditionally performing a first modular multiplication by multiplying a first multiplicand variable by a first multiplier variable followed by a modular operation and storing result of the first modular multiplication into a first variable;

unconditionally performing a second modular multiplication by multiplying a second multiplicand variable by a second multiplier variable followed by the modular operation and storing result of the second modular multiplication into the first variable; and

outputting a final result of the modular exponentiation from a memory location of the first variable in the computer system,

wherein the first variable, the first multiplicand variable and the second multiplicand variable are purposely arranged to be the identical variable.

2. (Currently Amended) [[A]]The method according to claim 1, wherein if said bit is said least significant bit of said exponent, said fourth value is initialized as 1wherein the first multiplier variable is respectively set to 1 and said digital message if a first examined bit is respectively equal to 1 and 0; the second multiplier variable is respectively set to 1 and said digital message if a second examined bit is respectively equal to 0 and 1; and the first and second examined bits are adjacent binary digits composing said secret exponent key.

3. (Currently Amended) [[A]]The method according to claim 2, wherein said bit is one bit of bits of said exponent and is a one or a zero further characterized by that no multiplier of any multiplication operation is arranged to store an intermediate result of any modular multiplication operation such that the method is resistant to M safe error attacks.

Application No. 10/615,065  
Amendment dated December 28, 2006  
Reply to Office Action of September 28, 2006

Docket No.: 4444-0294PUS1

4. (Cancelled)

5. (Currently Amended) An apparatus performing a cryptographic operation on a digital message, the apparatus comprising a computer program executing a modular exponentiation involving a secret exponent key, the computer program being implemented by a programming language source comprising codes executing the steps of: for protecting public key schemes from timing, power monitoring and fault attacks comprising:

means for obtaining a message for use in a cryptographic operation;

means for obtaining a modulus and an exponent corresponding to said cryptographic operation, wherein said exponent contains at least one bit;

means for initializing a first value as a one, and assigning said message to a second value;

means for executing a module exponentiation algorithm on each bit of said exponent from the most significant bit to the least significant bit, wherein said algorithm comprising:

means for input a bit into an inverter and storing the output as a third value, and assigning the next bit of said bit as a fourth bit;

means for determining whether said third value is a one or a zero;

means for updating said first value with the result of squaring if said third value is a one, modulo said modulus said first value, updating said first value with the result of multiplying, modulo said modulus said first value by said second value if said third value is a one;

means for determining whether said fourth value is a one or a zero; and

Application No. 10/615,065  
Amendment dated December 28, 2006  
Reply to Office Action of September 28, 2006

Docket No.: 4444-0294PUS1

~~means for updating said first value with the result of squaring, modulo said modulus said first value if said fourth value is a zero, updating said first value with the result of multiplying, modulo said modulus said first value by said second value if said fourth value is a one;~~

~~means for updating said bit with the next bit of said bit, and executing steps of said algorithm on said bit until said bit being said least significant bit of said exponent;~~

~~means for determining whether said bit is a one or a zero; and~~

~~means for storing and outputting said first value~~

unconditionally performing a first modular multiplication by multiplying a first multiplicand variable by a first multiplier variable followed by a modular operation and storing result of the first modular multiplication into a first variable;

unconditionally performing a second modular multiplication by multiplying a second multiplicand variable by a second multiplier variable followed by the modular operation and storing result of the second modular multiplication into the first variable; and

outputting a final result of the modular exponentiation from a memory location of the first variable in the apparatus.

wherein the first variable, the first multiplicand variable and the second multiplicand variable are purposely arranged to be the identical variable.

6. (Currently Amended) [[An]] The apparatus according to claim 5, wherein said bit is one bit of bits of said exponent and is a one or a zero wherein the first multiplier variable is respectively set to 1 and said digital message if a first examined bit is respectively equal to 1 and 0; the second multiplier variable is respectively set to 1 and said digital message if a second

Application No. 10/615,065  
Amendment dated December 28, 2006  
Reply to Office Action of September 28, 2006

Docket No.: 4444-0294PLJS;

examined bit is respectively equal to 0 and 1; and the first and second examined bits are adjacent binary digits composing said secret exponent key.

7. (Currently Amended) [[An]] The apparatus according to claim 5, wherein said inverter is used for output a one if input a zero into it, and output a zero if input a one into it further characterized by that no multiplier of any multiplication operation is arranged to store an intermediate result of any modular multiplication operation such that the apparatus is resistant to M safe error attacks.

8. (Currently Amended) A computer-readable medium containing a computer program performing a cryptographic operation on a digital message, the computer program, when executed in a computer system, executing a modular exponentiation involving a secret exponent key, the computer program being implemented by a programming language source comprising codes executing the steps of: for protecting public key schemes from timing, power monitoring and fault attacks containing logic code that executing the steps of:

obtaining a message for use in a cryptographic operation;  
obtaining a modulus and an exponent corresponding to said cryptographic operation,  
wherein said exponent contains at least one bit;  
initializing a first value as a one, and assigning said message to a second value;  
executing a module exponentiation algorithm on each bit of said exponent from the most significant bit to the least significant bit, wherein said algorithm comprising the steps of:

Application No. 10/615,065  
Amendment dated December 28, 2006  
Reply to Office Action of September 28, 2006

Docket No.: 4444-0294PUS1

input a bit to an inverter and storing the output as a third value, and assigning the next bit of said bit as a fourth bit;

if said third value is a zero, updating said first value with the result of squaring, modulo said modulus said first value, if said third value is a one, updating said first value with the result of multiplying, modulo said modulus said first value by said second value; and

if said fourth value is a zero, updating said first value with the result of squaring, modulo said modulus said first value, if said fourth value is a one, updating said first value with the result of multiplying, modulo said modulus said first value by said second value;

updating said bit with the next bit of said bit, and executing steps of said algorithm on said bit until said bit being said least significant bit of said exponent; and

storing and output said first value

unconditionally performing a first modular multiplication by multiplying a first multiplicand variable by a first multiplier variable followed by a modular operation and storing result of the first modular multiplication into a first variable;

unconditionally performing a second modular multiplication by multiplying a second multiplicand variable by a second multiplier variable followed by the modular operation and storing result of the second modular multiplication into the first variable; and

outputting a final result of the modular exponentiation from a memory location of the first variable in the computer system,

wherein the first variable, the first multiplicand variable and the second multiplicand variable are purposely arranged to be the identical variable.

Application No. 10/615,065  
Amendment dated December 28, 2006  
Reply to Office Action of September 28, 2006

Docket No.: 4444-0294PUS1

9. (Currently Amended) [[A]] The computer-readable medium according to claim 8, wherein if said bit is said least significant bit of said exponent, said fourth value is initialized as 1  
wherein the first multiplier variable is respectively set to 1 and said digital message if a first  
examined bit is respectively equal to 1 and 0; the second multiplier variable is respectively set to  
1 and said digital message if a second examined bit is respectively equal to 0 and 1; and the first  
and second examined bits are adjacent binary digits composing said secret exponent key.

10. (Currently Amended) [[A]] The computer-readable medium according to claim 9,  
wherein said inverter is used for output a one if input a zero into it, and output a zero if input a  
one into it further characterized by that no multiplier of any multiplication operation is arranged  
to store an intermediate result of any modular multiplication operation such that the computer  
system is resistant to M safe error attacks.

11. (Cancelled)